A Numerical Slow Manifold Approach to Model Reduction for Optimal Control of Multiple Time Scale ODE

Dirk Lebiedz*and Marcel Rehberg†

February 8, 2013

Abstract

Time scale separation is a natural property of many control systems that can be exploited, theoretically and numerically. We present a numerical scheme to solve optimal control problems with considerable time scale separation that is based on a model reduction approach that does not need the system to be explicitly stated in singularly perturbed form. We present examples that highlight the advantages and disadvantages of the method.

1 Introduction

Optimization based control in practice depends on accurate models with small prediction error and computation of a (feedback) control that is close or at least consistent with the true optimal control for the process under consideration [23]. Often the desired accuracy can only be provided by nonlinear large scale models which leads to problems in online control, for example via nonlinear model predictive control (NMPC), [8, 5] due to the computational demand. Model reduction therefore plays an important part in the development of control systems, see the paper by Marquardt [23] who gives a concise review of model development and model reduction techniques.

Model reduction can be divided into model order reduction which aims at decreasing the dimension of the state space and model simplification which tries to simplify the evaluation of the model equations. Both approaches can be combined and essentially strive to capture the most important features of the dynamic process at the cost of an error in the reduced compared to the full model. The trade off between lost accuracy and benefits of the reduced model always has to be considered and carefully balanced depending on the application at hand.

2 Model Order Reduction

In the remainder of this article we will only refer to model order reduction. Therefore we introduce the system

$$\dot{\tilde{z}} = \tilde{f}(\tilde{z}, u),\ \tilde{z}(0) = \tilde{z}_0 \tag{1}$$

with state $\tilde{z}(t) \in \mathbb{R}^n$ and control $u(t) \in \mathbb{R}^m$. The right hand side $\tilde{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is assumed to be in C^∞. A general approach to model order reduction can be summarized in the following steps [23]:

*Institute for Numerical Mathematics, Ulm University, Helmholtzstr. 20, 89081 Ulm, Germany
†Institute for Numerical Mathematics, Ulm University, Helmholtzstr. 20, 89081 Ulm, Germany

1. Find a diffeomorphism $T : \mathbb{R}^n \to \mathbb{R}^n$ that maps $\tilde{z}$ via

$$\tilde{z} - \tilde{z}^* = T(z) \Leftrightarrow z = \tilde{z}^* + T^{-1}(\tilde{z}),$$

onto the new state $z(t) \in \mathbb{R}^n$, where $\tilde{z}^*$ is a possibly nonzero set point. The aim of this coordinate change is to separate directions in the phase space of (1) that have strong contributions to the dynamics from those that only contribute in a minor way.

2. Decompose the new state space into $x(t) \in \mathbb{R}^p$ and $y(t) \in \mathbb{R}^q$ such that $z = (x, y)^{\mathrm{T}}$ and $n = p + q$. Here x will play the role of the dominant states.

3. Assemble new dynamic systems for x and y from

$$\dot{z} = (\mathrm{D}_z T(z))^{-1} \tilde{f}(\tilde{z}^* + T(z), u)$$

and obtain

$$\begin{aligned} \dot{x} &= f(x, y, u), \quad x(0) = x_0, \\ \dot{y} &= g(x, y, u), \quad y(0) = y_0. \end{aligned}$$

The smoothness of the right hand sides $f : \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^m \to \mathbb{R}^p$ and $g : \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^m \to \mathbb{R}^p$ is determined by the smoothness of T and T^{-1}.

4. Eliminate the dynamic equation for y by one of the following methods:

Truncation: Set $y = 0$ for the reduced dynamic system

$$\dot{\tilde{x}} = f(\tilde{x}, 0, u), \quad \tilde{x}(0) = x_0$$

with $\tilde{x} \approx x$ and state space dimension m.

Residualization: Set $\dot{y} = 0$ to obtain the differential-algebraic system

$$\begin{aligned} \dot{\tilde{x}} &= f(\tilde{x}, \tilde{y}, u), \quad \tilde{x}(0) = x_0, \\ 0 &= g(\tilde{x}, \tilde{y}, u). \end{aligned}$$

The dimension of the model is not reduced.

Slaving: Obtain a map $\tilde{y} = \phi(x, u)$ either from the residualization approach by solving the algebraic equation explicitly or through an independent method. Using

$$\dot{\tilde{x}} = f(\tilde{x}, \phi(\tilde{x}, u), u), \quad \tilde{x}(0) = x_0$$

leads to a reduced model with state space dimension m.

Several model order reduction methods have been proposed in the past and we proceed to give a short description of some of them.

Nonlinear balancing is an analytical method based on the theory of nonlinear Hankel operators and their attributed singular value functions aimed at obtaining a nonlinear map T [9]. In practice, empirical balancing that incorporates samples of the systems behavior for different inputs and initial values can be used [10]. In that case the map T is linear.

Proper orthogonal decomposition (POD) is based on sampling representative trajectories of (1), called snapshots [13]. Similarly to balancing a linear transformation matrix T is obtained using singular value decomposition. For a focus in the context of optimal control see [15].

Additional approaches include combinations of balancing and POD [16] and moment matching for nonlinear systems [1].

2.1 Slow Invariant Manifolds

Our approach to model order reduction is based on time scale separation which is a frequent feature of complex dynamic processes. A theoretical environment for such system is provided in singular perturbation theory [11] where we deal with a system of the form

$$\begin{aligned} \dot{x} &= f(x, y, u, \varepsilon), \quad x(0) = \xi(\varepsilon), \\ \varepsilon\dot{y} &= g(x, y, u, \varepsilon), \quad y(0) = \eta(\varepsilon). \end{aligned} \tag{2}$$

The parameter ε is assumed to be small ($0 < \varepsilon \ll 1$) and reflects the time scale separation. The fast modes y evolve on the time scale $\mathcal{O}(\varepsilon^{-1})$ whereas the slow state dynamics are $\mathcal{O}(1)$. We put forward the following assumptions:

A1 The involved functions f, g, ξ, and η are at least $R + 2$ times continuously differentiable with respect to their arguments on their respective domains of interest.

A2 Let $\varepsilon = 0$ in (2), then the reduced system is given by

$$\begin{aligned} \dot{x} &= f(x, y, u, 0), \quad x(0) = \xi(0), \\ 0 &= g(x, y, u, 0), \quad y(0) = \eta(0). \end{aligned}$$

There exist solutions $x(t) = x_0(t)$ and $y(t) = y_0(t)$ of the reduced system for $t \in [0, T]$.

A3 The Jacobian

$$g_y = \mathrm{D}_y\, g(x_0(t), y_0(t), u(t), 0) \in \mathbb{R}^{q\times q}$$

has $1 \le k \le q$ eigenvalues λ_i, $i = 1, 2, \ldots, k$ with $\Re(\lambda_i) < -\mu$ and $q - k$ eigenvalues with $\Re(\lambda_i) > \mu$, $i = k + 1, k + 2, \ldots, q$ where $\mu > 0$. In other words the Jacobian has no purely imaginary eigenvalues and there is at least one stable direction.

From the third condition it follows that g_y is nonsingular, so the algebraic equation $0 = g(x, y, u, \varepsilon)$ can (at least locally) always be solved with respect to y. Let the assumptions A1–A3 hold, then there exists ([11], Theorem 1) an ε_0 and a k-dimensional manifold $S(\varepsilon)$ such that the solution of (2) can be expanded into series representations for $\varepsilon < \varepsilon_0$ provided that $\eta(\varepsilon) \subset S(\varepsilon)$:

$$\begin{aligned} x(t, u, \varepsilon) &= x^*(t, u, \varepsilon) + X(t/\varepsilon, u, \varepsilon), \\ y(t, u, \varepsilon) &= y^*(t, u, \varepsilon) + Y(t/\varepsilon, u, \varepsilon) \end{aligned}$$

with

$$\begin{aligned} x^*(t, u, \varepsilon) &= \sum_{r=0}^{R} x_r^*(t, u)\varepsilon^r + \mathcal{O}(\varepsilon^{R+1}), \quad y^*(t, u, \varepsilon) = \sum_{r=0}^{R} y_r^*(t, u)\varepsilon^r + \mathcal{O}(\varepsilon^{R+1}), \\ X(t/\varepsilon, u, \varepsilon) &= \sum_{r=0}^{R} X_r(t/\varepsilon, u)\varepsilon^r + \mathcal{O}(\varepsilon^{R+1}), \quad Y(t/\varepsilon, u, \varepsilon) = \sum_{r=0}^{R} Y_r(t/\varepsilon, u)\varepsilon^r + \mathcal{O}(\varepsilon^{R+1}). \end{aligned}$$

The fast motions are captured in the so called boundary layer corrections $X(t/\varepsilon, u)$ and $Y(t/\varepsilon, u)$ which converge to 0 exponentially fast. For the purpose of model order reduction we neglect the boundary layer correction and focus on the slow or outer solution $x^*(t, u, \varepsilon)$. A central result is the following that goes back to Fenichel [7], see also [12], and is related to the geometric singular perturbation approach to the problem.

Theorem 1 ([12], Theorem 2.1, Fenichel, asymptotically stable slow manifolds)**.** *Let assumptions A1-A3 hold. Then, for any sufficiently small ε, there is a function h that is defined on a compact domain $K \subset \mathbb{R}^q \times \mathbb{R}^m$ such that the graph*

$$\mathcal{M}_\varepsilon = \{(x, y) \,|\, y = h(x, u, \varepsilon),\ (x, u) \in K\}$$

is locally invariant under (2)*. The function h admits an asymptotic expansion,*

$$h(x, u, \varepsilon) = \sum_{r=0}^{R} h_r(x, u)\varepsilon^r + \mathcal{O}(\varepsilon^{R+1}). \tag{3}$$

The manifold $\mathcal{M}_\varepsilon$ is also called slow invariant manifold (SIM). Utilizing $h(x, u, \varepsilon)$ we can reduce the system (2) to

$$\dot{x}^* = f(x^*, h(x^*, u, \varepsilon), u, \varepsilon), \quad x(0) = \xi(\varepsilon),$$

in accordance to the slaving approach introduced earlier. In practice only an approximation to h can be feasibly computed. Using h_0 corresponds to setting $\varepsilon = 0$ in (2) and solving the algebraic system for y, the fast states are assumed to be relaxed immediately.

Using an explicit formula or the SIM for model order reduction presumes that the system is given in singularly perturbed form and therefore the method skips step 1 of the algorithm discussed earlier. For general systems (1) a nonlinear coordinate transformation T would have to be explicitly known to apply the outlined theory. For systems where the small parameter ε can be identified, conditions on when such a transformation exists and how it might be constructed are given in [22]. Otherwise a state space decomposition into fast and slow modes has to be based on physical insight or numerical methods. Among them are eigenvalue analysis of the linearized system or singular value analysis of sensitivity matrices [18].

2.2 Approximation of the SIM

To approximate the SIM, in general only a numerical procedure will be feasible either because the analytic computations to obtain the coefficients for the asymptotic expansion (3) of h can not be carried out explicitly or the system can not be transformed to the singular perturbed form. In this section we therefore regard the general system

$$\begin{aligned} \dot{x} &= f(x, y, u), \\ \dot{y} &= g(x, y, u). \end{aligned}$$

We assume that the state space decomposition into fast modes x and slow modes y is already carried out either by employing a priori knowledge or by using one of the methods mentioned above. Our approach [21, 17, 28, 19] is based on optimization of trajectory pieces or points in the state space where the slow variables are fixed at a certain point in time t^* and the according fast states are computed in dependence of the slow variables, i.e. the SIM is parametrized by the slow states and can represented by a function smooth function $y = h(x, u)$. The underlying idea is that the fast states will relax onto the SIM as fast as the system dynamics possibly allow and then stay on it. A parametrization for u also has to be chosen. We will use a piecewise constant control function later in a multiple shooting approach to solve the optimal control problem, therefore we will use $u = \text{const}$ here. The optimization problem for the computation of

a reduced model is

$$\begin{aligned} &\min_y \Phi(x,y,u) \\ \text{subject to: } &\dot{x} = f(x,y,u) \\ &\dot{y} = g(x,y,u) \\ &x(t^*) = x^*, \quad u = u^*. \end{aligned} \tag{4}$$

Let F be the full right hand side vector, hence

$$F(x,y,u) = \begin{pmatrix} f(x,y,u) \\ g(x,y,u) \end{pmatrix}$$

and J be the Jacobian of the full dynamic system with respect to x and y, i.e.

$$J = \mathrm{D}_{x,y} F(x,y,u)$$

For Φ we will either use

$$\Phi(x,y,u) = \int_0^{t^*} \|JF\|_2^2 \,\mathrm{d}t. \tag{5}$$

or

$$\Phi(x,y,u) = \|JF\|_2^2 . \tag{6}$$

In the first objective the value of x is fixed at the end of the integration interval because the fast modes are unstable in backward time. This means that trajectories starting at points $y^* = y(t^*)$ that are not on the manifold will exponentially move away from it and thus a large contribution to the objective function is created. In the second case the dynamic equations are no longer constraints of the optimization problem.

Remark 1. $\|JF\|_2^2$ in both objective functionals can be linked to minimizing curvature in the phase space [19] and a variational principle [21]. There is a relation to the zero derivative principle [37].

Additionally, for the application of $h(x,u)$ in optimal control we also need at least first order sensitivities $\mathrm{D}_x h(x,u)$ and $\mathrm{D}_u h(x,u)$. However, they can be easily obtained from the KKT system at the solution point y^* of the optimization problem (4) [30].

Numerically, for the integral based objective (5) either single shooting or collocation is used to obtain a nonlinear program (NLP) which is subsequently solved with an interior point method implemented in the software package `IPOPT` [35]. The local formulation (6) is solved with a general Gauß-Newton method specifically tailored to the problem at hand [30]. In all cases warm starts are employed to improve convergence of the optimization if the problem has to be solved for a series of fixed values (x^*, u^*).

3 Optimal Control and Reduced Models

Singularly perturbed optimal control problems have been studied extensively in the past [14, 34, 26, 6]. The main idea is to decompose the full system into a slow and a fast part and infer properties and solutions of the full problem from the independent analysis of the two subproblems. The linear case is well understood, but for nonlinear systems the situation is much more difficult and only for special cases useful explicit results can be obtained. To highlight some of the

problems and what we can expect at most from our approach we review shortly the nonlinear state regulator problem [27]:

$$\begin{aligned} &\min_u j(u) = E(x(1), \varepsilon y(1), \varepsilon) + \int_0^1 f_0(x, y, u, \varepsilon) \mathrm{d}t \\ \text{subject to: } &\dot{x} = f(x, y, u, \varepsilon), \quad x(0) = x_0, \\ &\varepsilon \dot{y} = g(x, y, u, \varepsilon), \quad y(0) = y_0, \\ &x(t) \in \mathbb{R}^p, \quad y(t) \in \mathbb{R}^q, \quad u(t) \in \mathbb{R}^m. \end{aligned} \tag{7}$$

For convenience all functions are supposed to be C^∞ functions of their arguments on any domain of interest. The $\varepsilon y(1)$ in the final time cost avoids E to depend on the fast variable y for $\varepsilon = 0$. Using the Pontryagin minimum principle with the Hamiltonian

$$\mathcal{H}(x, y, \lambda_x, \lambda_y, u, \varepsilon) = f_0 + \lambda_x^{\mathrm{T}} f + \lambda_y^{\mathrm{T}} g$$

we get (additionally to the primal dynamic equations) the following ODE system for the adjoint variables λ_x and λ_y:

$$\begin{aligned} \dot{\lambda}_x &= -\mathrm{D}_x \mathcal{H}, \quad \lambda_x(1) = \mathrm{D}_x E(x(1), \varepsilon y(1), \varepsilon), \\ \varepsilon \dot{\lambda}_y &= -\mathrm{D}_y \mathcal{H}, \quad \lambda_x(1) = \varepsilon \mathrm{D}_y E(x(1), \varepsilon y(1), \varepsilon). \end{aligned} \tag{8}$$

We note that there are no restrictions on the value of the control u thus $\mathrm{D}_u \mathcal{H} = 0$ is a necessary condition for a minimum to occur. Moreover we assume the strong Legendre-Clebsch condition, $\mathrm{D}_{uu} \mathcal{H}$ is positive definite, to hold. In that case a (locally) optimal control $u(t)$ that minimizes the cost functional $j(u)$ exists [4] for $\varepsilon > 0$. It also allows to solve $\mathrm{D}_u \mathcal{H} = 0$ (locally) for $u = \omega(x, y, \lambda_x, \lambda_y, \varepsilon)$ and replace it in (7) and (8). We now have a singularly perturbed boundary value problem. The main challenge with problems of this type is to determine a reasonable reduced problem, i.e. setting $\varepsilon = 0$ in (7) and (8). In general not all boundary values can be satisfied and some of them have to be relaxed. In this case the choice is obvious: $y(0)$ and $\lambda_y(1)$, the boundary values associated with the fast modes can not be fulfilled because for $\varepsilon = 0$ their values are determined by algebraic equations. Even if a reduced problem can be stated, for nonlinear problems it is not generally possible to postulate conditions for a solution to exist. Besides the smoothness assumptions the following restrictions are necessary.

A1' The reduced problem

$$\begin{aligned} \dot{x} &= f(x, y, \omega, 0), \quad x(0) = x_0, \\ \dot{\lambda}_x &= -\mathrm{D}_x \mathcal{H}(x, y, \lambda_x, \lambda_y, \omega, 0), \quad \lambda_x(1) = \mathrm{D}_x E(x(1), 0, 0), \\ 0 &= g(x, y, \omega, 0), \\ 0 &= -\mathrm{D}_y \mathcal{H}(x, y, \lambda_x, \lambda_y, \omega, 0), \end{aligned}$$

has a unique solution $x^0(t)$, $y^0(t)$, $\lambda_x^0(t)$, and $\lambda_y^0(t)$.

A2' The Jacobian

$$\mathcal{H}_y = \begin{pmatrix} \mathrm{D}_{\lambda_y y} \mathcal{H} & \mathrm{D}_{\lambda_y \lambda_y} \mathcal{H} \\ -\mathrm{D}_{yy} \mathcal{H} & -\mathrm{D}_{y \lambda_y} \mathcal{H} \end{pmatrix} \in \mathbb{R}^{2q \times 2q}$$

evaluated along $x^0(t)$, $y^0(t)$, $\lambda_x^0(t)$, and $\lambda_y^0(t)$ has no purely imaginary eigenvalues, moreover we require it to have exactly q eigenvalues with positive and q eigenvalues with negative real part.

For the problem at hand it turns out that $\mathcal{H}_y$ is block diagonal with two identical blocks with opposite sign and symmetric and therefore the second condition on $\mathcal{H}_y$ is automatically fulfilled if all eigenvalues have nonzero real part. The second condition guarantees the solvability of the algebraic part of the reduced system and the stability of the boundary layer corrections. If A1' and A2' hold, the solution of the full problem converges to the solution of the reduced problem for $\varepsilon \to 0$ and the following series representations can be stated [27, 11]:

$$\begin{aligned} x(t,\varepsilon) &= x^*(t,\varepsilon) + X_L(t/\varepsilon,\varepsilon) + X_R(s/\varepsilon,\varepsilon), \\ y(t,\varepsilon) &= y^*(t,\varepsilon) + Y_L(t/\varepsilon,\varepsilon) + Y_R(s/\varepsilon,\varepsilon), \\ u(t,\varepsilon) &= u^*(t,\varepsilon) + U_L(t/\varepsilon,\varepsilon) + U_R(s/\varepsilon,\varepsilon), \\ j(u) &= j^*(t,\varepsilon) + J_L(t/\varepsilon,\varepsilon) + J_R(s/\varepsilon,\varepsilon), \end{aligned} \tag{9}$$

with $s = 1 - t$. Boundary layer corrections emerge at both ends of the time interval and appropriate series representations can be found to all right-hand-side terms in (9). The eigenvalue condition on $\mathcal{H}_y$ is necessary for the stability of the left and right hand boundary layer corrections to be stable in forward backward time, respectively.

The manifold $h(x,u,\varepsilon)$ (3) is an intrinsic property of a singularly perturbed system and it exists independently from the use of the system as constraint of an optimal control problem. Hence it can be used to reduce the dimension of the optimal control problem (7) by replacing y and we find

$$\begin{aligned} \min_u j(u) &= E(x(1),\varepsilon h(x(1),u(1),\varepsilon),\varepsilon) + \int_0^1 f_0(x,h,u,\varepsilon)\mathrm{d}t \\ \text{subject to: } \dot{x} &= f(x,h,u,\varepsilon), \quad x(0) = x_0, \\ & x(t) \in \mathbb{R}^p, \quad u(t) \in \mathbb{R}^m. \end{aligned}$$

Its solution will correspond to $x^*(t,\varepsilon)$, $u^*(t,\varepsilon)$, and $j^*(\varepsilon)$ which means we have no way of obtaining information about the boundary layer corrections in this case. If the manifold is only an approximation of order k with respect to its ε series representation the solutions of the reduced problem will also be approximations of order k.

4 Numerical Solution of the Optimal Control Problem

We will provide a short overview of the numerical methods we use to solve general optimal control problems of the type

$$\begin{aligned} &\min_{x,u,T,p} j(x,u,p) \\ \text{subject to: } &\dot{x} = f(x,u,r) \\ &e(t,x,u,T,r) = 0 \\ &i(t,x,u,T,r) \le 0 \end{aligned}$$

where $x(t) \in \mathbb{R}^m$ is the state, $u(t) \in \mathbb{R}^r$ is the control, $T \in \mathbb{R}$ is the final time, and $r \in \mathbb{R}^s$ are parameters. We do not discuss the solvability of the problem and conveniently assume that local solutions exist. In order to solve the problem numerically we have to discretize the control and state functions $u(t)$ and $x(t)$, respectively. For this purpose we use multiple shooting [3], which means we divide the overall time interval $[0,T]$ into N subintervals with node points t_k, $k = 0,1,\ldots,N$, $t_k < t_{k+1}$, $t_0 = 0$, $t_N = T$. On each interval the control is kept constant with values $u_k \in \mathbb{R}^m$. This constant input is used to solve the ODE for $x(t)$ on each interval with the

help of a numerical integration routine. Introducing x_k^0, $k = 1, 2, \ldots, N$ as initial values for the dynamic equation on each interval with states $x_k(t)$, $t \in [t_{k-1}, t_k]$ and u_k as value of the control function we can formulate a finite dimensional nonlinear program

$$\begin{aligned}
&\min_{x_k^0, u_k, T, p} \sum_{k=1}^{N} j(x_k, u_k, p) \\
\text{subject to: } &\dot{x_k} = f(x_k, u_k, p), \quad k = 1, 2, \ldots, N, \\
&x_k(t_k) - x_{k+1}^0 = 0, \quad k = 1, 2, \ldots, N-1, \\
&e(t, x_k, u_k, T, p) = 0, \quad t \in [t_{k-1}, k],\ k = 1, 2, \ldots, N, \\
&i(t, x_k, u_k, T, p) \leq 0 \quad t \in [t_{k-1}, k],\ k = 1, 2, \ldots, N.
\end{aligned}$$

The equality constraints $x_k(t_k) - x_{k+1}^0$ ensure the continuity of the solution at the multiple shooting nodes. The method is implemented as a C++ program using `IPOPT` [35] for solving the NLP, CppAD, a tool for automatic differentiation [2] for obtaining accurate derivatives in the NLP as well as for a BDF integrator [32] that is used to solve the ODEs on the multiple shooting intervals and provide sensitivities.

5 Evaluation of the Manifold

Eventually, we have to evaluate the manifold map $y = h(x, u)$ for arbitrary points $(x, u) \subset \mathbb{R}^p \times \mathbb{R}^m$. We do not regard ε as argument of h here, since it is either a fixed parameter for the numerical solution of the optimal control problem in case of singularly perturbed problems or we regard general problems without explicit dependence on a small parameter ε. We will discuss two alternatives: Solving the model reduction problem online, i.e. whenever an evaluation of $h(x, u)$ is needed while solving the optimal control problem, and interpolation of offline precomputed data obtained by evaluating $h(x, u)$ on a discrete set of points $\mathcal{C} \subset \mathbb{R}^p \times \mathbb{R}^m$. Both methods have intrinsic advantages and disadvantages. The online method can be easily applied since no preparation steps have to be taken. However, calculating $h(x, u)$ is costly and involves the solution of a NLP which could slow down the overall computation. On the contrary the interpolation approach provides a direct, fast, and easy to evaluate object, that promises a larger speed up. However, it suffers from the need to precompute the manifold data and building the interpolation object, both tasks take a considerable amount of time. This makes this approach only effective if the optimal control problem has to be solved very often (e.g. in NMPC) so that the time spend in the preliminary stages is outweighed by the overall performance gain. Furthermore, especially for higher dimensional problems, the storage needed for the interpolation data might be to large to be handled properly for given hardware resources.

5.1 Online Evaluation

If an evaluation of $h(x, u)$ is needed for a certain (x_0, u_0) we solve the model reduction problem (4) with the slow states and control fixed to (x_0, u_0). For performance reasons only the local formulation (6) is feasible because integration of the full model is dispensed with and a general Gauß-Newton method can be used for efficient solution of the minimization problem [31, 30, 20]. Although it seems that this approach contradicts the purpose of model reduction, since the full right hand side still has to be evaluated, there is a computational advantage due to the decreased stiffness of the reduced model. In addition the points at which $h(x, u)$ has to be evaluated will typically be close to each other which makes it possible to use warm starts for the Gauß-Newton

procedure. Thus, in general, only very few iterations will be needed to solve the model reduction problem. This is in principle similar to solving an explicitly given differential-algebraic equation where usually inside the integration routine in each time step only a few iterations of a nonlinear equation solver are needed for the algebraic part of the dynamic problem.

5.2 Interpolation

The interpolation approach is based on an interpolation function $\hat{h}(x, u)$ that will be used instead of a pointwise computation of $h(x, u)$. We choose radial basis function (RBF) interpolation [36] because it is independent of the dimension of the input space and configuration of the interpolation nodes (i.e. grid free). The following short presentation of the subject is strongly based on [36]. Given a set of nodes $\mathcal{C} = \{x_k\}_{k=1}^N$, $\mathcal{C} \subset \Omega \subset \mathbb{R}^m$, radial basis interpolants are of the form

$$s(x) = \sum_{k=1}^{N} \lambda_k \Phi(x, x_k), \quad \lambda_k \in \mathbb{R}$$

with basis functions $\Phi : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$. The domain $\Omega \subset \mathbb{R}^m$ is assumed to be open and bounded and satisfy an interior cone condition. The interpolation is carried out by determining the coefficients λ_k such that

$$L_k(s) = L_k(f) = f_k, \quad f_k \in \mathbb{R}, \tag{10}$$

holds, where $f, s \in H$ are functions, H is a Hilbert space of functions $\mathbb{R}^d \to \mathbb{R}^l$ and $L_k \in H^*$ are linear functionals from the dual of H. The following property of a function Φ is central.

Definition 2 (Positive definite function)**.** A continuous function $\Phi : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ is said to be positive definite if for all $N \in \mathbb{N}$, all sets of pairwise distinct nodes $\mathcal{C} = \{x_k\}_{k=1}^N$, and all $\alpha \in \mathbb{R}^N \setminus \{0\}$ it holds that

$$\sum_{\ell=1}^{N} \sum_{k=1}^{N} \alpha_\ell \alpha_k \Phi(x_\ell, x_k) > 0.$$

Remark 2. Positive definite functions $\Phi(x, y)$ are also known as kernels and give rise to reproducing kernel Hilbert spaces, a topic we do not want to pursue any further here, see the book by Wendland [36] for more details.

In the most simple case of pointwise evaluation as the functionals in (10), i.e. $L_k(s) = \delta_{x_k} s = s(x_k)$, the interpolation condition becomes

$$\delta_{x_\ell} s = s(x_\ell) = \sum_{k=1}^{N} \lambda_k \Phi(x_\ell, x_k) = \delta_{x_\ell} f = f_\ell$$

and we obtain the linear system

$$A\lambda = F, \quad A_{\ell,k} = \Phi(x_\ell, x_k), \quad \lambda = \begin{pmatrix} \lambda_1 & \lambda_2 & \dots & \lambda_N \end{pmatrix}^{\mathrm{T}}, \quad F = \begin{pmatrix} f_1 & f_2 & \dots & f_N \end{pmatrix}^{\mathrm{T}},$$

with A positive definite since $\alpha^{\mathrm{T}} A \alpha > 0$ holds for all $\alpha \in \mathbb{R}^N \setminus \{0\}$ by definition. That guaranties a unique solution to the interpolation problem for all sets of pairwise distinct nodes.

The same approach can also used for Hermite interpolation where we interpolate not only the function value itself but also partial derivatives. The relevant functionals L_k are thus given by

$$L_k = \delta_{x_k} \circ \mathrm{D}^{\alpha_k}, \quad k = 1, 2, \dots, N, \quad \alpha_k \in \mathbb{N}^d,$$

where the differential operator D^{α_k} indicates concatenated derivatives according to the multi-index α_k. In general we demand that $x_k \neq x_\ell$ or $\alpha_k \neq \alpha_\ell$ for $k \neq \ell$ to guarantee linear independence of the functionals and therefore a unique solution to the interpolation problem. With our model reduction procedure for each interpolation node the function value and all partial derivatives of first order can be computed. If the interpolation nodes are chosen pairwise distinct than the linear independence follows. The interpolant is given by

$$s(x) = \sum_{k=1}^{N} \lambda_k \, \mathrm{D}_2^{\alpha_k} \, \Phi(x, x_k).$$

The subscript 2 of the differential operator indicates differentiation with respect to the second variable. The interpolation matrix has entries of the form

$$\mathrm{D}_1^{\alpha_\ell} \, \mathrm{D}_2^{\alpha_k} \, \Phi(x_\ell, x_k)$$

and it can be shown that for positive definite and sufficiently smooth Φ it is again also positive definite and thus the interpolation problem can be uniquely solved.

In practice univariate radial basis functions are commonly used, i.e. $\Phi(x, y) := \phi(\|x - y\|_2)$. We will use the Gaussian function

$$\phi(r) = \mathrm{e}^{-c^2 r^2}, \quad c \in \mathbb{R}, c > 0.$$

The parameter c is called the shape parameter. It plays an essential role for the interpolation error and the stability of the interpolation process by virtue of the fact that it is strongly connected to the condition of the interpolation matrix. In the case of pointwise interpolation the entries of A are $\mathrm{e}^{-c^2 \|x_\ell - x_k\|_2}$, $\ell, k = 1, 2, \ldots, N$ which, for $c \to 0$, will converge to 1 for all $x_\ell, x_k \in \mathbb{R}^m$. Conversely, for $c \to \infty$ the basis function ϕ will either converge to 0 if $x_\ell \neq x_k$ or to 1 if $x_\ell = x_k$. This means the matrix A tends to being singular in the first case and becoming the unit matrix in the second case. The interpolation error $\|s - f\|$ is also subject to the same trade of principle. Determining a "good" c is crucial for the performance of the interpolation. Therefore we use the algorithm suggested in [29] which is based on a computational favorable reformulation of a leave-one-out optimization scheme.

Since we want to use the interpolation in the optimization we are interested in a fast evaluation. We use a partition of unity approach to divide the domain of interest Ω into smaller subdomains and thereby bound the computing cost. To be more precise we look for a overlapping covering of Ω by open and bounded sets Ω_j, $j = 1, 2, \ldots, M$ and continuous functions $\omega_j(x) : \mathbb{R}^m \to \mathbb{R}$ such that

$$\sum_{j=1}^{M} \omega_j(x) = 1, \ \forall x \in \Omega \text{ and } \omega_j(x) = 0, \ \forall x \notin \Omega_j.$$

Assuming that we have a feasible covering $\{\Omega_j\}$ we can build local interpolants $s_j(x)$ for each Ω_j. Additionally let

$$I(x) = \{j | x \in \Omega_j\}$$

be an index function that returns the indices of the patches a point x is contained in. A global interpolant is then simply given by

$$s(x) = \sum_{j \in I(x)} \omega_j(x) s_j(x).$$

Under certain conditions on the covering and the ω_j it can be shown that the global interpolant enjoys the same error rates as in the naive global approach.

Two key ingredients are needed to really take computational advantage of the method: Partition Ω in a way so that all Ω_j contain about the same amount of node points and have the index function $I(x)$ be $\mathcal{O}(1)$ in terms of computing time, i.e. the cost of finding the patches a random point lies in does neither depend on the overall number of centers N nor on the number of patches M. In that case evaluation is $\mathcal{O}(1)$ also, since for any number of nodes we can partition Ω in a way that the number of points in a patch is below a certain constant threshold which means that the sums, that have to be evaluated for each patch have a constant number of terms which together with the constant time look-up leads to constant evaluation time.

If we assume the node points x_k to be uniformly distributed in Ω one practical way to achieve both aims is to use a fixed-grid structure which consists of axis parallel overlapping boxes. Because of the parallelism many operations, like index querying can be independently performed in each dimension. Given a set of nodes $\mathcal{C}$, overlap factor $\gamma \in (0, 0.5)$, and a lower bound for the average number of points in one box the number of boxes and their border coordinates can be computed. If o_i, $i = 1, 2, \ldots, d$ is the length of a box in dimension i the overlap factor γ determines the fraction of o_i by which two boxes overlap in dimension i, $\gamma < 0.5$ ensures that a point x can only be in maximal 2 boxes per coordinate direction.

The last missing part are the w_j. We choose dimension independent radial polynomials as suggested in [33] for the same purpose. They are given by

$$\omega_j(x) = \begin{cases} p \circ b_j(x) & x \in \Omega_j, \\ 0 & \text{else}, \end{cases}$$

with $p : \mathbb{R} \to \mathbb{R}$,

$$p(r) = -6r^5 + 15r^4 - 10r^3 + 1$$

a polynomial that fulfills the spline like conditions $p(0) = 1$, $p(1) = 0$, $\mathrm{D}^k\, p(0) = \mathrm{D}^k\, p(1) = 0$, $k = 1, 2$ and $b_j : \mathbb{R}^m \to [0, 1]$,

$$b_j(x) = 1 - \prod_{i=1}^{d} \frac{4(x_i - l_i^j)(r_i^j - x_i)}{(r_i^j - l_i^j)^2},$$

where $l^j \in \mathbb{R}^m$ and $r^j \in \mathbb{R}^m$ are the lower left and upper right corner coordinates of the box Ω_j, respectively. The polynomial p gives rise to a two times continuously differentiable global interpolant $s(x)$.

Besides function evaluation we at least also need first order partial derivatives during the optimization procedure which are computed by differentiating the interpolation function symbolically with respect to x_i.

6 Results

We present some numerical examples. Computation times given were obtained on an Intel Xeon E5620 (2.40 GHz) machine running 64 bit Debian Squeeze.

6.1 Enzym Kinetics

The first example is based on a simple Michaelis-Menten enzyme kinetics in singularly perturbed form [24]. The (full) problem is

$$\begin{aligned} &\min \int_0^5 -50y + u^2 \mathrm{d}t \\ \text{subject to: } &\dot{x} = -x + (x + 0.5)y + u, \\ &\varepsilon \dot{y} = x - (x + 1.0)y, \\ &x(0) = 1, \quad y(0) = y_0. \end{aligned} \tag{11}$$

The control and the objective are artificial and not related to a realistic model scenario. The reduced problem is obtained by replacing y with $h(x, u)$, eliminating the ODE for y and the initial condition y_0. The reduced problem is thus

$$\begin{aligned} &\min \int_0^5 -50h(x, u) + u^2 \mathrm{d}t \\ \text{subject to: } &\dot{x} = -x + (x + 0.5)h(x, u) + u, \\ &x(0) = 1. \end{aligned} \tag{12}$$

We set $\varepsilon = 10^{-2}$ and use the multiple shooting approach described above for numerical solution. To obtain initial values for x and y at the multiple shooting nodes we integrate the uncontrolled system ($u(t) = 0$) numerically starting at $x(0) = 0$ and $y(0) = y_0$. The overall time interval is divided into 40 equidistant multiple shooting intervals. The termination tolerance for `IPOPT` was set to 10^{-4} and the integration tolerance of the BDF-integrator to 10^{-6}. The discretized full problem has 204 variables whereas the reduced problem has 163. Lastly, bounds are introduced for the state variables and the control. We use the lower bounds $x_l = y_l = u_l = 0$ and the upper bounds $x_u = y_u = u_u = 5.5$ on all multiple shooting nodes.

The performance of the full system (11) depends on the initial value y_0. For $y_0 = 0$ we have an average runtime of 2.5 s and 26 NLP iterations. For $y_0 = 0.5$, which is the first order approximation $h_0(1, 0)$ we find 2.1 s for 22 NLP iterations and lastly for $y_0 = 1$ we get 2.2 s and 22 iterations.

Next we used the online computation of h. The algorithm clocks in at 2.1 s and 22 iterations, which means no performance gain compared to the full problem. However, the number of NLP iterations in the subproblem of approximating the manifold is interesting. The maximum is 4 iterations but 60 % of the calls terminate after 2 and 37 % only after 1 iterations. The call to the model reduction routine is by now done through an external library which produces a lot of overhead, for example in terms of right hand side function evaluations. Tight integration into the BDF integration algorithm might lead to a significant speed up.

For the interpolation approach one first needs to choose a reasonable set of nodes. Although the RBF method is grid independent for convenience we used a Cartesian grid on $[-0.5, -0.5] \times [10, 10]$. The performance of the interpolator depends of course on the number of points but also on the number of points per patch and overlap in the partition of unity approach. To assess the influence we scatter searched the region $\{20, 30, 35, 40\} \times \{0.025, 0.05, 0.1, 0.15\} \times \{5, 10, 15\}$ for points in each direction, overlap and points per patch respectively. Median runtime was 0.87 s and median number of NLP iterations 24.5. Moreover, the fastest combination took 0.61 s and only 21 iterations compared to the slowest which needed 1.4 s and 23 iterations. It is apparent that using the interpolation approach in this case is beneficial from an performance point of view. In the best case it is nearly 4 times faster than the full problem.

Table 1: Summary of various statistics concerning the solution of problem (11) and (12). The steps statistics refer to the integration and are the sums over all NLP iterations and multiple shooting intervals.

Problem	Time	NLP iter	Time per iter	Steps	Rej. steps
(11)	2.3 s	23	0.1 s	43 165	7369
(12) online	2.1 s	22	0.1 s	5520	1281
(12) offline (median)	0.9 s	24.5	0.04 s	-	-
(12) offline (best)	0.6 s	21	0.03 s	5194	1236

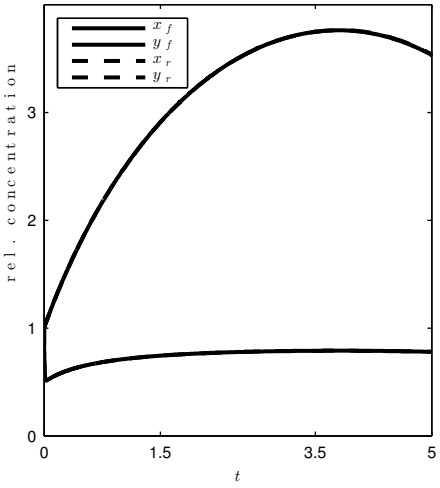

Figure 1: Example trajectories using the control from the full problem (11), subscript f and the reduced problem (12), subscript r. Both trajectories overlap.

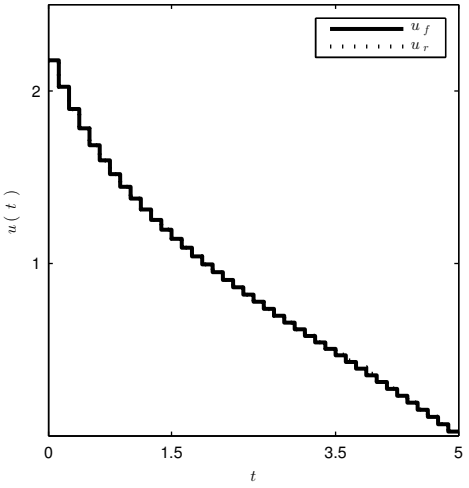

Figure 2: Example controls computed from the full problem (11), subscript f and the reduced problem (12), subscript r. Both controls overlap.

The main reason for the speed up is not so much the reduction in the number of optimization variables but mainly the reduced stiffness along the manifold h. In the integration routine larger step sizes are possible which greatly reduces the computational effort. This especially pays off in the multiple shooting approach since the initial values at the multiple shooting nodes are subject to optimization and they might be set away from the SIM for the full system in each iteration of the NLP solver leading to transient behavior of the fast trajectories on each interval and therefore forces the integrator to use small steps. An overview of example integrator statistics is given in table 1 as well as the result of the performance tests.

If we use the computed optimal control from the reduced system as input of the full system we obtain virtually the same objective values as for the control computed with the full system. In this example the error of the reduction is negligible which can also be concluded from the system itself. Example trajectories and controls are plotted in figure 1 and 2 respectively. In both plots the subscript f refers to the results obtained using the computed control from the full system (11), whereas r refers to the results using the control computed from the reduced problem (12).

6.2 Voltage Regulator

The next example is taken from [25], example 4.2. It describes a voltage regulator with 5 states governed by a set of linear ODEs. The problem is given by

$$\begin{aligned} \min \frac{1}{2}\int_0^2 x_1^2 + u^2 \mathrm{d}t \\ \text{subject to: } \dot{x}_1 &= -\frac{1}{5}x_1 + \frac{1}{2}x_2, \\ \dot{x}_2 &= -\frac{1}{2}x_2 + \frac{8}{5}y_1, \\ \varepsilon\dot{y}_1 &= -\frac{5}{7}y_1 + \frac{30}{7}y_2, \\ \varepsilon\dot{y}_2 &= -\frac{5}{4}y_2 + \frac{15}{4}y_3, \\ \varepsilon\dot{y}_3 &= -\frac{1}{2}y_3 + \frac{3}{2}u. \end{aligned} \tag{13}$$

The problem, although in essence linear-quadratic, has some interesting features: The coupling between the slow and fast subsystems is only through the one fast state y_1 which means that only one state has to be reproduced during the optimization, i.e. we are only interested in $y_1 = h(x_1, x_2, u)$. The reduced problem is

$$\begin{aligned} \min \frac{1}{2}\int_0^2 x_1^2 + u^2 \mathrm{d}t \\ \text{subject to: } \dot{x}_1 &= -\frac{1}{5}x_1 + \frac{1}{2}x_2, \\ \dot{x}_2 &= -\frac{1}{2}x_2 + \frac{8}{5}h(x_1, x_2, u). \end{aligned} \tag{14}$$

First, we set $\varepsilon = 0.2$ and solve both problems on 10 multiple shooting intervals with the `IPOPT` tolerance set to 10^{-3}. Again a selection of initial values for the full system was used. An overview is given in table 2. Note, that this selection leads to a set of two initial values for the reduced system, namely $x_0 = (-10, 0)^{\mathrm{T}}$ and $x_0 = (-10, 10)^{\mathrm{T}}$. The initial values for the state variables at the multiple shooting nodes are obtained through integrating the ode system with the initial control $u(t) = 0$. Bounds are introduced as follows: $y_1 \in [-20, 20]$, $y_2 \in [10, 50]$, $y_3, y_4, y_5 \in [-10^8, 10^8]$, and $u \in [-15, 15]$.

Using the control computed from the reduced problem for the full problem leads to extremely large objective values in comparison and thus renders the model reduction unusable in this case. This can also be seen in figures 3 and 4 which compares the full system with the reduced system solved with the online evaluation of $h(x, u)$. Additionally there is no runtime advantage: The full system needs on average 1.1 s compared to the offline approach which needs 1.4 s which is also the median timing of the online method.

If we increase the spectral gap by setting $\varepsilon = 2 \cdot 10^{-3}$ the results are much more favorable. First of all the computed input from the reduced model is very close to solution of the full problem. Therefore also the objective values are similar. See figures 5 and 6 for an example. With the reduced model we find a significant computational advantage, as documented in table 3. The runtime for the full problem depends strongly on the initial values and varies between 2.3 s to 5.1 s which is between 5 to 10 times slower compared to the fastest solution of the reduced problem with the offline method. The online approach is around 2 to 3 times faster. A further advantage worth mentioning in this context is that the time needed for the reduced problem is

Table 2: Final objective values of problems (13) and (14) for a selection initial values and $\varepsilon = 0.2$.

x_0	(13)	(14), online	(14), offline
$(-10, 0, 0, 0, 0)$	32.9	35.1	34.9
$(-10, 0, 10, 0, 10)$	25.1	521.7	612.7
$(-10, 0, 0, 10, 10)$	24.7	648.0	750.1
$(-10, 10, 10, 10, 10)$	20.4	782.6	830.3
$(-10, 10, 0, 0, 10)$	20.5	521.4	559.4
$(-10, 10, 10, 0, 10)$	20.0	579.1	619.5

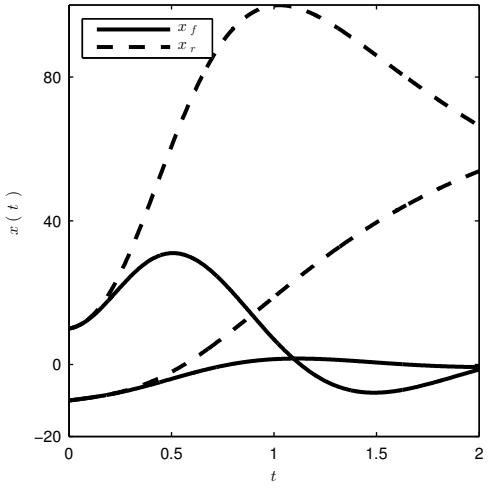

Figure 3: Example trajectories using the control from the full problem (13), subscript f and the reduced problem (14), subscript r with $\varepsilon = 0.2$.

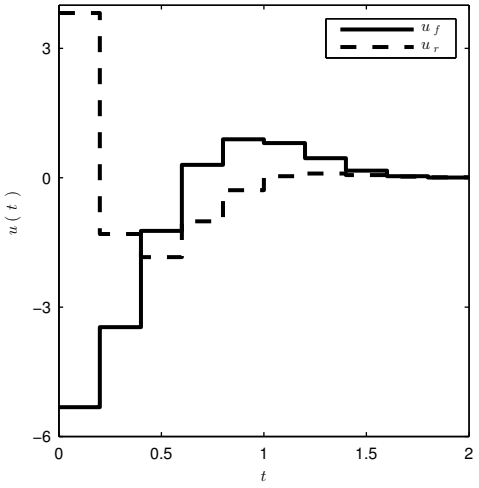

Figure 4: Example controls computed from the full problem (13), subscript f and the reduced problem (14), subscript r with $\varepsilon = 0.2$.

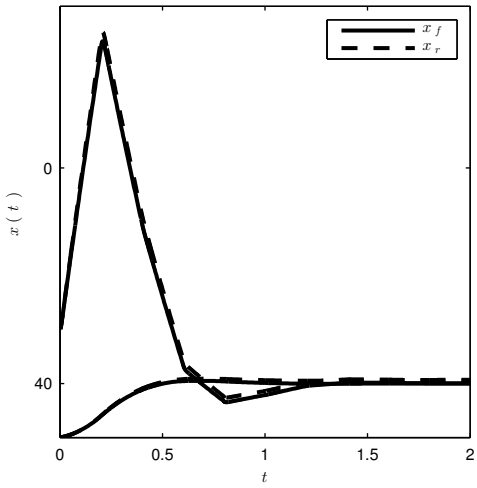

Figure 5: Example trajectories using the control from the full problem (13), subscript f and the reduced problem (14), subscript r with $\varepsilon = 2 \times 10^{-3}$.

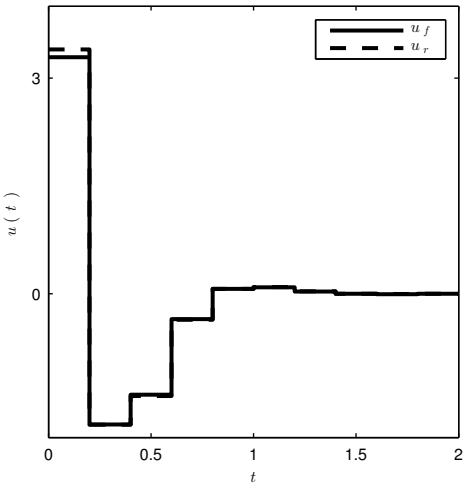

Figure 6: Example controls computed from the full problem (13), subscript f and the reduced problem (14), subscript r with $\varepsilon = 2 \times 10^{-3}$.

Table 3: Summary of various statistics concerning the solution of problem (13) and (14) for $\varepsilon = 0.2 \times 10^{-3}$. Timings are averages over all initial values.

Problem	Time	NLP iter	Time per iter
(13)	3.7 s	37.7	0.1 s
(12) online	1.6 s	16.5	0.1 s
(14) offline (median)	1.3 s	18	0.07 s
(14) offline (best)	0.5 s	16	0.03 s

much less dependent on the initial values, which makes the computation more reliable in online control scenarios where the next input has to be computed within a given time frame.

As in the enzyme example we systematically tried various parameter combinations (points per dimension, overlap and points per patch) for the interpolator. We already mentioned the best and median runtime values, however it should also be noted that bad parameter combinations can decrease the algorithmic performance significantly. The maximum time needed for both ε values and sets of initial values was over 16 s. In a considerable number of cases the problem could not even be solved. This shows that the interpolator approach has to be tuned carefully but further analysis reveals that at least in this case the best configuration is the same for both initial values and ε.

7 Concluding Remarks

Given an optimal control problem, the aim of model reduction is to determine and solve a smaller problem and use its solution as input to the full scale problem hoping for computational benefits while still obtaining a feasible and nearly optimal control. Our approach to reduce the dimension

of the state space is based on time scale separation, i.e. processes evolving on slow and fast time scales within the same system. According to [17, 21, 28, 30] we formulate a nonlinear optimization problem that identifies a slow manifold in the state space, parametrized by the slow states. This manifold hence defines a map $y = h(x, u)$ of the slow variables and control onto the fast variables and can be used to reduce the dynamic system.

Singular perturbation theory delivers a framework for optimal control problems involving fast/slow differential systems. Using the Pontryagin minimum principle one arrives at a singularly perturbed boundary value problem. Its solution consists of three components: A slowly varying part which represents the system confined to the slow manifold and two fast vanishing boundary layer corrections. Because we use an approximation to the slow manifold to represent the fast states and thereby reduce the dimension of the state space we are only able to obtain an approximation to the slow part of the optimal control solution. Two examples, both of which are singularly perturbed systems, are used to illustrate that if the boundary layer corrections are small, the solution of the reduced system can produce controls that drive the full system in a nearly optimal fashion. Solving the reduced problem is up to ten times faster if the offline, interpolation based method is used for the manifold map h. The numerical scheme presented here can be used unmodified to solve general nonlinear optimal control problems, i.e. problems not explicitly in singular perturbed form.

If larger systems, especially with more slow states are considered the interpolation approach will suffer from the curse of dimensionality because of the exponentially growing number of nodes and with it also interpolation data that has to be handled. This problem could to a certain degree be overcome by using a more suited data structure (e.g. kd-trees) and by using a tight, problem specific state space and control domain and take advantage of the ability of using scattered nodes.

The online method is not subject to the dimensionality problem, however its evaluation for one input point takes considerably longer since the full system has to be evaluated in the general Gauß-Newton method. Still the benefit from reduced stiffness can speed up the overall solution of the optimal control problem. A tight integration into the ODE integration routine, similar to a DAE solver would greatly decrease unnecessary overhead and increase speed and stability.

References

[1] A. Astolfi. Model reduction by moment matching for linear and nonlinear systems. *Automatic Control, IEEE Transactions on*, 55(10):2321 –2336, oct. 2010.

[2] Bradley M. Bell. Automatic differentiation software cppad., 2010.

[3] Hans Georg Bock and Karl J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings of the Ninth IFAC World Congress, Budapest*. Pergamon, Oxford, 1984.

[4] AE Bryson and YC Ho. *Applied Optimal Control: Optimization, Estimation and Control*. Taylor and Francis, London, 1975.

[5] M. Diehl, H. G. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002.

[6] M. Dmitriev and G. Kurina. Singular perturbations in control problems. *Automation and Remote Control*, 67(1):1–43, January 2006.

[7] Neil Fenichel. Geometric singular perturbation theory for ordinary differential equations. *Journal of Differential Equations*, 31:53–98, 1979.

[8] Rolf Findeisen, Lars Imsland, Frank Allgöwer, and Bjarne A. Foss. State and output feedback nonlinear model predictive control: An overview. *European Journal of Control*, 9(2–3):190–207, 2003.

[9] K. Fujimoto and J. Scherpen. Balanced realization and model order reduction for nonlinear systems based on singular value analysis. *SIAM Journal on Control and Optimization*, 48(7):4591–4623, 2010.

[10] Juergen Hahn and Thomas F. Edgar. An improved method for nonlinear model reduction using balancing of empirical gramians. *Computers & Chemical Engineering*, 26(10):1379 – 1397, 2002.

[11] Frank Hoppensteadt. Properties of solutions of ordinary differential equations with small parameters. *Communications on Pure and Applied Mathematics*, 24:807–840, 1971.

[12] Hans G. Kaper and Tasso Joost Kaper. Asymptotic analysis of two reduction methods for systems of chemical reactions. *Physica D*, 165:66–93, 2002.

[13] Gaetan Kerschen, Jean-Claude Golinval, Alexander F. Vakakis, and Lawrence A. Bergman. The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview. *Nonlinear Dynamics*, 41:147–169, 2005. 10.1007/s11071-005-2803-2.

[14] Petar V. Kokotović. Applications of singular perturbation techniques to control problems. *SIAM Review*, 26(4):501–550, 1984.

[15] Karl Kunisch and Stefan Volkwein. Proper orthogonal decomposition for optimality systems. *ESAIM: Mathematical Modelling and Numerical Analysis*, 42(01):1–23, 2008.

[16] Sanjay Lall, Jerrold E. Marsden, and Sonja Glavaški. A subspace approach to balanced truncation for model reduction of nonlinear control systems. *International Journal of Robust and Nonlinear Control*, 12(6):519–535, 2002.

[17] Dirk Lebiedz. Computing minimal entropy production trajectories: An approach to model reduction in chemical kinetics. *Journal of Chemical Physics*, 120(15):6890–6897, April 2004.

[18] Dirk Lebiedz, Julia Kammerer, and Ulrich Brandt-Pollmann. Automatic network coupling analysis for dynamical systems based on detailed kinetic models. *Physical Review E*, 72(4):041911, October 2005.

[19] Dirk Lebiedz, Volkmar Reinhardt, and Jochen Siehr. Minimal curvature trajectories: Riemannian geometry concepts for slow manifold computation in chemical kinetics. *Journal of Computational Physics*, 229(18):6512–6533, September 2010.

[20] Dirk Lebiedz and Jochen Siehr. A continuation method for the efficient solution of parametric optimization problems in kinetic model reduction. arXiv:1301.5815, 2013.

[21] Dirk Lebiedz, Jochen Siehr, and Jonas Unger. A variational principle for computing slow invariant manifolds in dissipative dynamical systems. *SIAM Journal on Scientific Computing*, 33(2):703–720, 2011.

[22] R. Marino and P.V. Kokotovic. A geometric approach to nonlinear singularly perturbed control systems. *Automatica*, 24(1):31 – 41, 1988.

[23] W. Marquardt. Nonlinear model reduction for optimization based control of transient chemical processes. In J.W. Eaton J. B. Rawlings, B.A. Ogunnaike, editor, *Chemical Process Control VI, Tuscon, Arizona, 7-12.1.2001*, number 326, pages 12–42, 2002.

[24] J. D. Murray. *Mathematical Biology I: An Introduction*. Springer, 1993.

[25] D. Subbaram Naidu. *Singular Perturbation Methodology in Control Systems*. Institution of Engineering and Technology, 1988.

[26] D. Subbaram Naidu. Singular perturbations and time scales in control theory and applications: an overview. *Dynamics of Continuous, Discrete and Impulsive Systems, Series B: Applications and Algorithms*, 9:233–278, 2002.

[27] R. O'Malley. Singular perturbations and optimal control. In W. Coppel, editor, *Mathematical Control Theory*, volume 680 of *Lecture Notes in Mathematics*, pages 170–218. Springer Berlin / Heidelberg, 1978. 10.1007/BFb0065317.

[28] Volkmar Reinhardt, Miriam Winckler, and Dirk Lebiedz. Approximation of slow attracting manifolds in chemical kinetics by trajectory-based optimization approaches. *Journal of Physical Chemistry A*, 112(8):1712–1718, 2008.

[29] Shmuel Rippa. An algorithm for selecting a good value for the parameter c in radial basis function interpolation. *Advances in Computational Mathematics*, 11:193–210, 1999. 10.1023/A:1018975909870.

[30] Jochen Siehr. *Numerical optimization methods within a continuation strategy for the reduction of chemical combustion models*. PhD thesis, Ruprecht-Karls-Universität Heidelberg, Heidelberg, Germany, 2012.

[31] Jochen Siehr and Dirk Lebiedz. An optimization approach to kinetic model reduction for combustion chemistry. *Flow, Turbulence and Combustion*, page submitted, 2012.

[32] Dominik Skanda. *Robust optimal experimental design for model discrimination of kinetic ODE systems*. PhD thesis, University of Freiburg, Freiburg im Breisgau, Germany, 2012.

[33] Ireneusz Tobor, Patrick Reuter, and Christophe Schlick. Efficient reconstruction of large scattered geometric datasets using the partition of unity and radial basis functions. In *WSCG*, pages 467–474, 2004.

[34] A. B Vasil'eva and M. G. Dmitriev. Singular perturbations in optimal control problems. *Journal of Mathematical Sciences*, 34:1579–1629, 1986.

[35] Andreas Wächter and Lorenz T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[36] Holger Wendland. *Scattered Data Approximation*. Cambridge University Press, 2005.

[37] Antonios Zagaris, C. William Gear, Tasso Joost Kaper, and Yannis G. Kevrekidis. Analysis of the accuracy and convergence of equation-free projection to a slow manifold. *ESAIM: Mathematical Modelling and Numerical Analysis*, 43(4):757–784, 2009.